並列計算機PACS-32によるGauss-Jordan並列解法(数値計算のアルゴリズムの研究)

AUTHOR(S):

白川, 友紀; 上村, 健; 星野, 力

数理解析研究所講究録
483巻 1983年 131-146

並列計算機PACS-32によるGauss-Jordan並列解法

筑波大構造工学系　白川友紀（Tomonori Shirakawa）
筑波大　情報学類　上村　健（Takeshi Kamimura）
筑波大構造工学系　星野　力（Tsutomu Hoshino）

1. はじめに　PACS[1)2)]は，その名（Processor Array for Continium Simulation）の示すように，連続体すなわち近接作用系のシミュレーション，言い換えれば，偏微分方程式系の陽解法や反復法による数値計算のための超高速計算機のプロトタイプとして試作されたものである。この目的に合わせてPACS-32は32台のProcessing Unit（PU）を，8×4の直角格子状に並べ，各PUは上下左右に隣接する4台のPUおよびControl Unitと呼ばれるマイクロコンピュータとだけ直接にデータ交換可能であるような構造となっている。そのため逆に，このような事情が強調されるあまり，隣接しないPU間のデータ交換が必要とされる連立方程式の求解，マトリックスの演算，FFTなどにはPACSは無能であるかのように思われていた。[3)]少なくともPUの数をさらに多く

しても，PU間のデータ交換のための時間が増大し，計算速度の向上は頭打ちになると考えられていた。

ここでは，PACS-32において，離れたPU間のデータ交換が必要なガウスジョルダン法を用いて実際に連立一次方程式の求解を行なった結果，相当の性能が得られ，またPUの台数を増加した場合には台数に比例して計算速度が向上するという結果が得られたので紹介する。

2. PACS-32の構成

図1にPACS-32のシステム構成

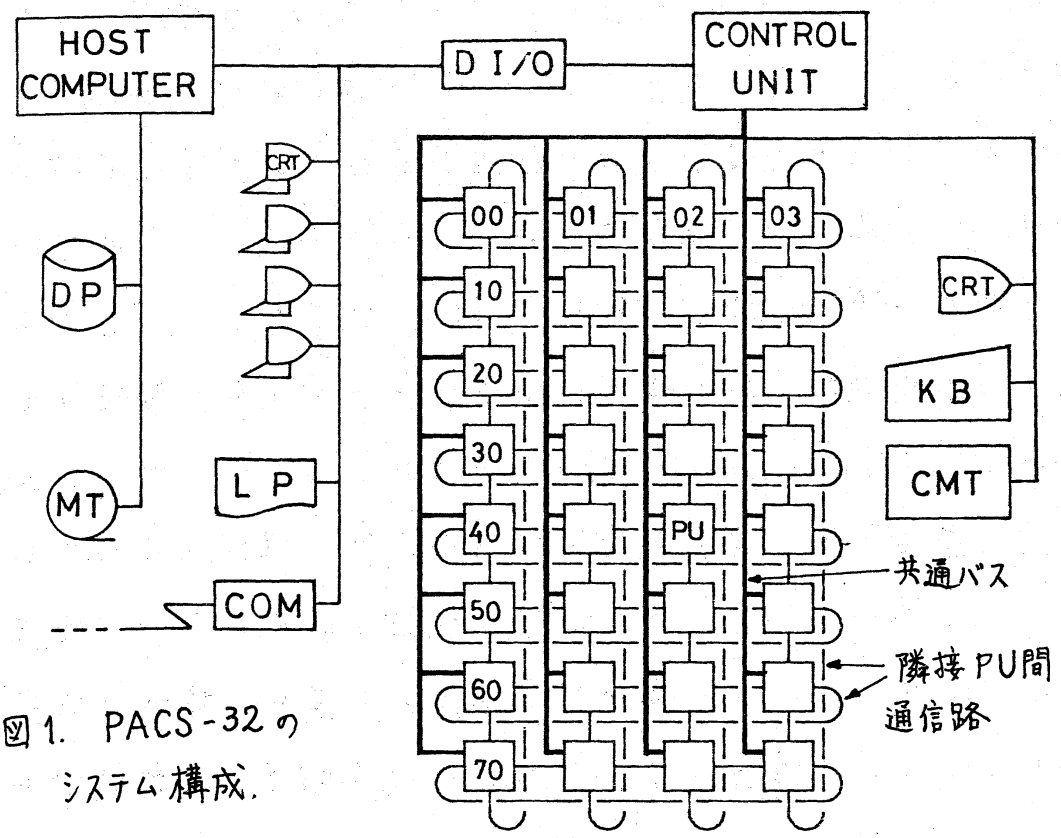


図1. PACS-32のシステム構成.

を示す。プログラムはHOST COMPUTER（汎用ミニコン）で作成し，データと共にCU，共通バスを通じてPUにロードする。隣接しないPU間でデータの交換を行なうには次の2つの方法を用いることができる。

i) 伝送（Routing）方式：左隣のPUから受け取ったデータを右隣へ回すというように順次受け渡しをしてゆき，離れたPUまでデータを伝える。

ii) CUを経由する方式：一旦，あるPUのデータをCUに転送し，CUから任意のPUに送る方式で，CUから全PUに放送（Broadcast）することもできる。

3. 効率について　並列計算機の方式の性能を評価するためには，並列化の効率を議論する必要があるが，計算機の効率の定義がはっきりしていないので，まずここで議論する効率の意味について述べる。

科学技術用計算機のハードウェアの演算能力を表わすために，FLOPSという単位を用いる。これは，その計算機の演算装置が1秒間に行なうことができる浮動小数点演算の回数を表わしたものである。PACS-32の場合は，各PUで使用している浮動小数点演算用のICの加算と乗算に要する時間の平均が約65μ秒であるのでPU 1台あたりの演算能力は

15～16 K FLOPS であり，32台全部では 約 0.5 M FLOPS であると言える。 しかし，今，計算量が 1 MFLOP すなわち求解に100万回の浮動小数点演算を必要とする問題が与えられ，これをPACS-32で解くとすると 1MFLOP／0.5MFLOPS＝2秒で解けるかと言うと，実際はそうはいかない。

i) PACSの場合，浮動小数点演算用ICとメモリとの間のデータ，計算結果のやりとりを含めると実効的な速度は1PUあたり 8～9 KFLOPS 程度になる。つまり，0.5～0.6 ぐらいの効率がかかることになる。 このような効率の低下は，演算装置とメモリとの間の速度差の大きな計算機では深刻であるのでこの効率低下を防ぐため様々の工夫がなされている。

ii) プログラム中でループやサブルーチンを使用すれば，ループ回数の計数や終了の判定，また引数の受渡しや復帰などにも時間がかかる。これらはコンパイラの性能などにもよるが，PACSの場合1つのPUの実効演算能力は約6KFLOPSぐらいに減る。

以上 i), ii) に述べた効率の低下は1つのPUでの処理速度に関するもので，従来の逐次処理方式の計算機にも存在したものである。 PACS-32では，さらに32台のPUで並列演算を行なった時の効率を考える必要がある。 1台のPUで計算を行なった時の32倍の速度が得られた時，効率100%

となるが，実際には以下のような効率低下が起こる。

iii) プロセッサ遊休：32台のPU全部に計算を割り当てないようなプログラムを作成すれば，遊ぶPUができ効率が低下する。プログラミングの容易さを得るためにユーザは多少の効率の低下は無視してPUを遊ばせることもある[4]が，その得失については一概に論ずることはできないし，効率の評価も難しいので本稿では問題としない。

iv) PU間のデータ転送：1台のPUで逐次処理を行なう時には必要が無いが，複数のPUで並列処理を行なう時にはPU間の通信，データ転送が必要となることがある。このような場合には，PU間通信，データ転送の時間は無駄時間と見做す。また，データ転送のためにPU間の同期をとるなどの必要があれば，そのための時間も実質的計算時間とは見做さない。ある並列処理の実行時間が全部で T_p であり，その内，データ転送とそれに付随する無駄時間が T_c であったとすると，この効率は，$\alpha_c = (T_p - T_c) / T_p$ で表わされる。

v) 同期待ち：PUの計算の進行をそろえるため，同期をとることがある。ある同期時から次の同期時までの間，各PU間の計算量が不均一である場合には早く計算を終えたPUは遊ぶことになる。全並列処理ののべ時間を T_p，同期待ちののべ時間を T_s とすると，この効率は $\alpha_s = (T_p - T_s) / T_p$

で表わされる。

vi) アルゴリズム：逐次処理において最高速であるアルゴリズムが並列計算においても最高速であるとは限らない。先に述べた効率 α_c や α_s が大きくなるようなアルゴリズムの方が計算量は多くとも，並列処理においては高速となることがある。並列処理に用いたアルゴリズムが，1 PU による逐次処理のアルゴリズムと異なる場合，その効率を α_A とする。α_A は，並列処理に用いたアルゴリズムにより，1 PU で処理した場合の実行時間 T_1 と，他の通常用いられる高速のアルゴリズムにより 1 PU で処理した場合の実行時間 T_A を用いて，$\alpha_A = T_A / T_1$ で表わされる。

以上述べたことをまとめて，本稿では並列化効率 α を次のように定義することにする。すなわち，$\alpha = T_A / (T_P \times P)$. ただし T_A は通常用いられるアルゴリズムにより 1 PU で処理した場合の実行時間，T_P は並列処理による実行時間，P は PU 台数である。iv), v), vi) で述べた要因以外に並列処理の効率低下の原因は考えられないので，$\alpha = \alpha_A(\alpha_c + \alpha_s - 1)$ となる。従って PACS-32 の実効演算能力は 約 6 KFLOPS × 32 × $\alpha \fallingdotseq 0.2\alpha$ MFLOPS となる。PACS-32 は将来 PU 数を 1万 ないし 100万個にすることを想定した実験機であるので

単に α が大きいだけでなく，α の PU 台数依存性が問題である。すなわち PACS 方式が大規模計算に対し有効であるためには，PU 数を増加させてもまったく α が減少しないか，あるいは PU 数が充分大きくなるまで α が減少しないことが必要である。

4. ガウスジョルダン法とガウス消去法について

係数行列が密であるような連立一次方程式を解く標準的アルゴリズムにガウス消去法があり，ここで採りあげたガウスジョルダン法にくらべ計算量が約 2/3 と少なくて済む。実際 PACS において 32 元の連立一次方程式を 1 つの PU の逐次処理によって解いた時，その計算時間はガウス消去法によれば約 8 秒，ガウスジョルダン法によれば 約 11 秒であった。しかし，32 台の PU により並列処理を行なった場合は，ガウス消去法によると 0.38 秒（内，前進消去に 0.33 秒），ガウスジョルダン法によると 0.33 秒であった。結局，ガウスジョルダン法とガウス消去法の前進消去の時間が同じで，後退代入の時間だけガウスジョルダン法の方が速いという結果になった。このことより並列処理にはガウスジョルダン法がむいていると言える。以上で示した実行時間には，ピボット選択の時間は含まれていない。　ところで，ガウスジョル

ダン法の場合の並列化による速度向上比は $11/0.33 \fallingdotseq 33.3$ となり PU 台数 32 を上回っている。これは実は，1台の PU で 32 元の連立方程式を解く時，2次元の配列を使いたいのであるが，PACS-32 のコンパイラ SPLM は2次元配列をサポートしていないのでデータの操作に時間がかかるためである。従って以上のデータからはアルゴリズム効率 $\alpha_A = 8/11$ を得るにとどめ，α_c，α_s の値は実際の並列処理中のデータ転送や，同期待ちに要する時間から調べることにする。

5. PACS-32 によるガウスジョルダン並列解法

32 台の PU を用いて 32 元の連立一次方程式を解く場合を例に並列ガウスジョルダン法[5]による解法を説明する。
問題が，

$$\begin{cases} a_{1,1}x_1 + a_{1,2}x_2 + \cdots\cdots + a_{1,32}x_{32} = a_{1,33} \\ a_{2,1}x_1 + a_{2,2}x_2 + \cdots\cdots + a_{2,32}x_{32} = a_{2,33} \\ \qquad\qquad \vdots \\ a_{32,1}x_1 + a_{32,2}x_2 + \cdots\cdots + a_{32,32}x_{32} = a_{32,33} \end{cases}$$

で与えられているとする。この時，32 台の PU に各々1行の係数と定数項（合計 33 個）の値をデータとして与え，部分ピボット選択法（partial pivotting）とガウスジョルダン法による計算ののちに各 PU に解の1つが得られるものとする。

求解の手順をPADにより図2に示す。連立方程式の求解には，

〔ステップⅠ〕まずピボット選択を行なう。すなわち $a_{1,1}$，$a_{2,1}$，…，$a_{32,1}$ の中から絶対値が最大のものを選ぶ。

〔ステップⅠ-1〕各PU (p) が持っている係数 $a_{p,1}$ のコピーを右隣のPUへ転送する。その後，左隣から転送されてきた係数 $a_{p-1,1}$ と自分の係数 $a_{p,1}$ を比較し，絶対値の大きい方 MAX ($a_{p,1}$, $a_{p-1,1}$) を選ぶ。この様子を図3(a)に示す。

〔ステップⅠ-2〕次に比較の結果記憶した絶対値の大きい方の係数の値を2つ右隣のPUへ順次送りにより伝送する。その後，2つ左隣から送られてきた値と先に自分が選んだ値 MAX ($a_{p,1}$, $a_{p-1,1}$) を比較し，絶対値の大きい方を選ぶ。これにより，横方向4つのPUの持つ係数の $a_{p,1}$ のうち，絶対値最大のものが4つのPU全部で選ばれたことになる。(図3(b))。

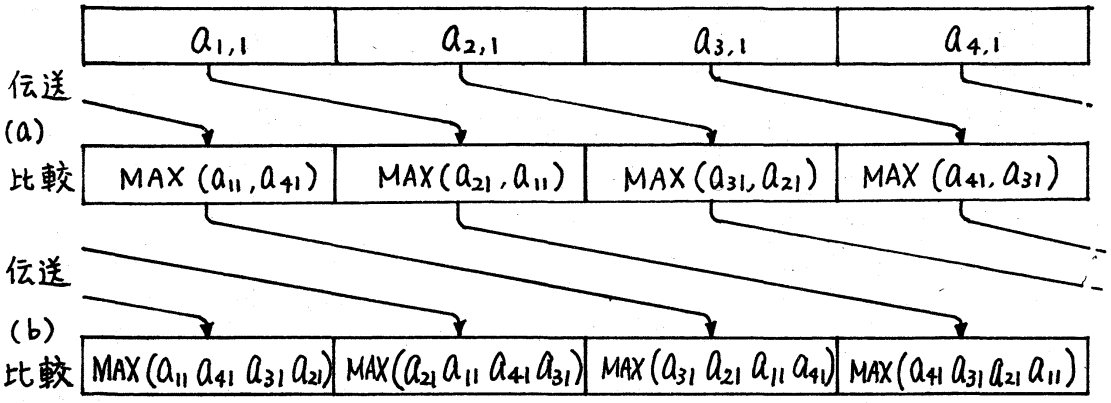
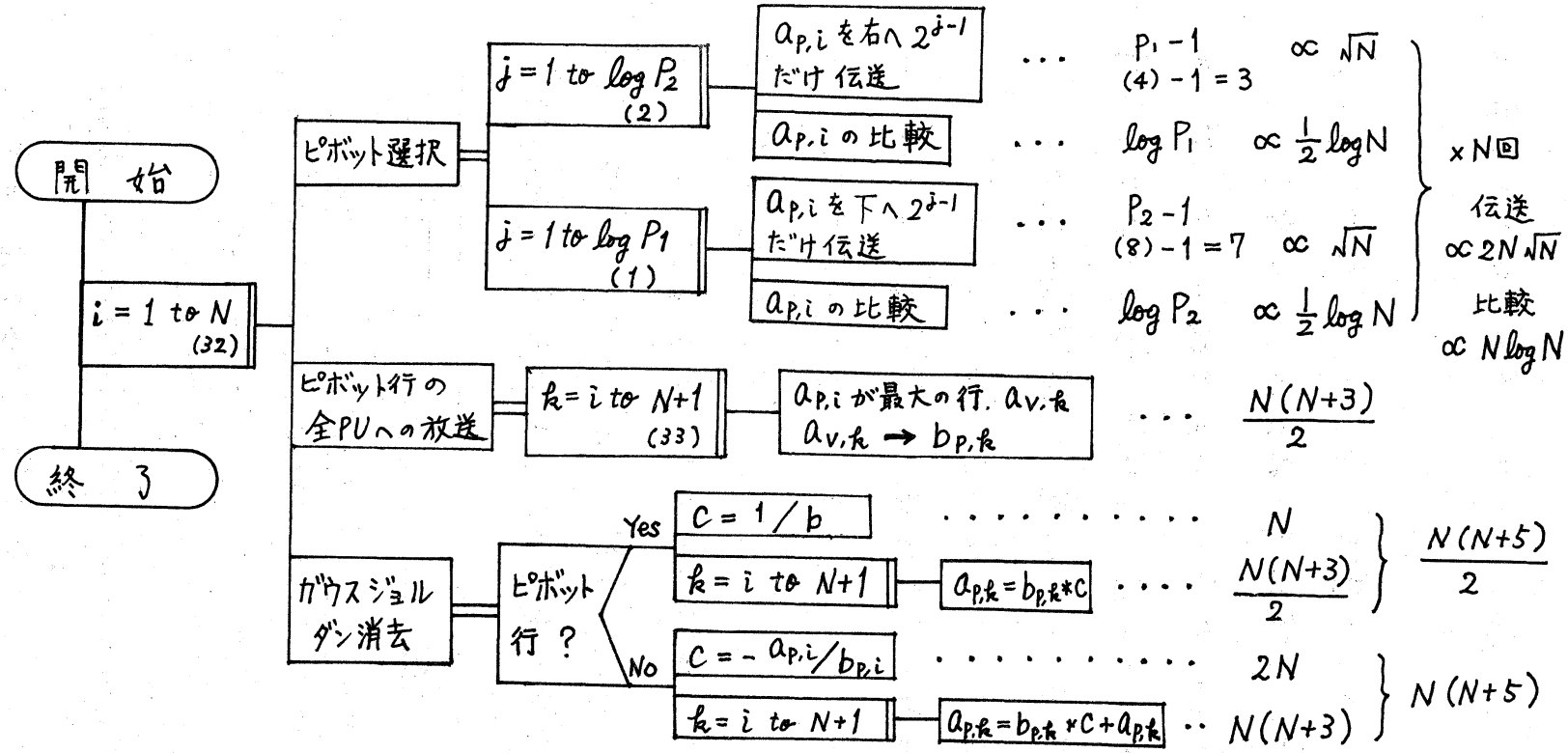


図3．伝送(Routing)とカスケード比較による最大値の求め方．



N：連立方程式の次元数　　P, P1, P2：プロセッサ(PU)台数
P = P1 × P2 の2次元直角格子配列
PACS-32の場合. P=32, P1=8, P2=4.
P∝N, P1, P2 ∝ √N

p：連立方程式の行番号

v：ピボット行の番号　　log：2を底とする対数.

図2. 並列ガウスジョルダン法による連立方程式の求解手順と各段階の処理量.

ステップI-1, I-2 で必要とされる処理量は 1PU あたり距離 1+2=3 の伝送(Routing)と2回の係数の比較(カスケード比較)である。この処理量も図2のPAD の右側に示す。以後,各ステップ毎に処理量を図2に示す。

〔ステップI-3〕 ステップI-1, I-2 と同様のことを上下方向についても行なう。すなわち,まず PU 1つ分の距離だけ下にデータを伝送し,比較し,大きい方を下に PU 2つ分の距離伝送し,比較し,さらに下に PU 4つ分送り大きい方を選ぶ。これにより 32 台の PU のもつ係数のうち絶対値最大のものが全PU に知れる。

〔ステップII〕 次に消去計算を行なう.

〔ステップII-1〕 ステップIで求めた $a_{1,1}, a_{2,1}, \cdots, a_{32,1}$ の内絶対値最大の係数 $a_{v,1}$ を持つ行をピボット行とするのでこの行の係数および定数項 $a_{v,1}, a_{v,2}, \cdots a_{v,33}$ を CU 経由で全 PU に放送(Broadcast)する。

〔ステップII-2〕 ピボット行を受け持っている PU では, $a_{v,1}, a_{v,2}, \cdots, a_{v,33}$ を $a_{v,1}$ で割る。 その結果 $a_{v,1}=1$ となる。ピボット行以外の行に対しては, $a_{p,1}, a_{p,2}, \cdots, a_{p,33}$ に対し, $a_{p,k} = a_{p,k} - a_{v,k} \times a_{p,1} / a_{v,1}$ $(k=1,2,\cdots,33)$ の計算を行なう。その結果 $a_{p,1}$ は 0 となる。

〔ステップIII〕 既にピボット行となった行については,ス

ステップⅠの操作においてピボット行となる行の候補からはずし，ステップⅠ，ステップⅡの操作を第2列，第3列，……，第32列の係数についてくり返し行なう。結果として各行および各列の係数が唯一つだけ1に，他がすべて0になり，元定数項であった $a_{1,33}$, $a_{2,33}$, ……, $a_{32,33}$ に解 x の値が求まる。

6. 並列処理の効率の評価　連立一次方程式の並列ガウスジョルダン法による求解の各処理の計算量，データ転送量が図2に示されている。これらの処理量は連立方程式の次元数 N の関数の形および $N=32$ の時の値で表わされている。ピボット選択に必要な処理量は $N=32$ の場合，Routing に320（距離×回数），比較に160回である。またBroadcastするデータ量は $33+32+\cdots+2=560$，消去計算は，ピボット行では592 FLOP，ピボット行以外では1行あたり1184 FLOPである。このピボット行とそれ以外の行の計算量の差からPUに同期待ちが生じる。この効率 α_s を図2に示された処理量から，Routing，比較，Broadcastにそれぞれ1処理あたり1FLOPの計算量と等しい時間がかかると仮定して求めると，

$$\alpha_s = \frac{(320+160+560)\times 32 + 1184\times 31 + 592}{(320+160+560)\times 32 + (1184\times 32)} \fallingdotseq 0.99$$

となり，Nの関数として表わせば，

$$\alpha_s = \frac{(2N\sqrt{N} + N\log N + \frac{N(N+3)}{2})N + N(N+5)(N-1) + \frac{N(N+5)}{2}}{(2N\sqrt{N} + N\log N + \frac{N(N+3)}{2})N + N(N+5)\cdot N}$$

$$= \frac{3N^2 + 4N\sqrt{N} + 2N\log N + 12N - 5}{3N^2 + 4N\sqrt{N} + 2N\log N + 13N}$$

となる．従って，Nが大きくなると α_s はさらに1に近づくので．以後 α_s はほぼ1と見做す。

α_c については，Routing と Broadcast に必要な時間が逐次処理には無く，並列化のために必要となった処理であるので，全処理時間からこれらのための所要時間を差引いた時間（純計算時間）を全処理時間で割れば求めることができる。図2から，純計算の中では消去計算に要する計算量のオーダが $O(N^2)$ で最も大きく，データ転送の中では Broadcast に要する処理量が同じく $O(N^2)$ で最も大きい。従ってNが大きくなれば効率 α_c は消去計算に要する時間と Broadcast に要する時間の比で決まる一定値に近づくと推測される。次節では，実際に連立一次方程式の求解を実行してその実行時間とデータ転送時間の実測により α_c を求める。

7. 実行結果とスケーリング則

図2に示した手順に従い，実際にPACS-32を用いて連立

方程式の部分ピボット選択と並列ガウスジョルダン法による求解を行なった。 PU台数より大きな次元数の連立方程式の場合には，1台のPUに2行以上の処理を受け持たせ，8, 16, 32, 64, 96, 128, 160元の連立方程式を解いた。各実行時における全処理時間とRoutingとBroadcastに要した時間を除いた純計算時間との比から効率α_cを求めた。結果を表1に示す。

使用PU台数	連立方程式の次元数	全処理時間(秒)	純計算時間(秒)	効率 α_c
4	32	1.9	1.7	0.90
8	8	0.074	0.037	0.50
8	32	1.15	0.89	0.77
16	16	0.21	0.11	0.52
16	32	0.81	0.51	0.63
32	32	0.59	0.32	0.54
32	64	2.6	1.7	0.67
32	96	6.7	5.2	0.77
32	128	14	12	0.86
32	160	25	22	0.88

表1. 部分ピボット選択付ガウスジョルダン法による連立一次方程式の求解時間と効率

表1の結果より，狭義の並列化効率α_cは，少なくとも，

0.5以上であり，PU台数を連立方程式の次元数と同時に増してゆくとわずかずつ増加し，また1PUあたりの受け持ち行数（N/P）を2，3，…… と増やすと効率 α_c もかなり大きくなることがわかった。 この関係を，各処理に必要な時間を調べて簡単に式で表わすことにより，スケーリング則を求めた。 1PUが受け持つ連立方程式の行数 $n = N/P$ をパラメータにすると，PとNを同時に大きくした時，効率 α_c は漸近的に，

$$\alpha_c = \frac{n}{n + 0.51}$$

に近づく。

Nが100以上になると効率 α_c はほとんど飽和し，上式の値に近くなる。 このスケーリング則により，PACS方式は，PU台数を増せば，効率が下がるどころか更に上がり，並列処理が有効に行なえることが明らかとなった。

7. おわりに

PACS-32において，ガウスジョルダン法を用いて連立一次方程式の求解を試み，その結果，PU台数を増しても高い並列化効率が得られることがわかった。これはRoutingとBroadcastの技術によるもので，同様の方法により，他の非隣接PU間のデータ転送を必要とするような問題にも適用できる可能性が示されたと考える。 今後，さら

にいろいろな応用分野への適用可能性を研究する必要があると考えられるので，このシステムの名を PACS から PAX に改名した。[6] PAX とは Processor Array for eXperiment の略であり，また，「平和」を意味するところからこのシステムが平和的利用に供されることを希望して付けたものである。

参考文献

1) 星野 力 「並列計算機のもたらす技術計算へのインパクト」 日本原子力学会誌，Vol. 22, No. 4 PP. 204-212 (1980)

2) 星野 力 「並列計算機 PACS による数値シミュレーション」 「数値計算のアルゴリズムの研究」研究集会報告集 P.19, (1981)

3) 「科学技術計算用コンピュータにおける高速化手法」 日経エレクトロニクス 8-3, PP. 122-136, (1981)

4) 星野 力 「並列計算機 PACS-32 による BWR 炉心計算」 日本原子力学会誌 . Vol. 23, No. 8 PP. 54-62, (1981)

5) DON HELLER "A SURVEY OF PARALLEL ALGORITHMS IN NUMERICAL LINEAR ALGEBRA" SIAM Review, Vol. 20, No. 4, Oct. PP. 740-777, (1978)

6) T. Hoshino 他, " Super Freedom Simulator PAX " The 16th IBM Computer Science Symp. Working Conf. Oct. 1-3, (1982) 箱根.